Name : Deep Pawar
CWID : A20545137
Sub : SOP Assignment 3 (CS536)

Date
Page

Q. 1.

a) Let $S \equiv$ if $x > y \rightarrow x := x - 1 \;\square\; x > y \rightarrow y := y + 1 \;\square\; x + y = 4 \rightarrow x := y / x \;\square\; x + y = 4 \rightarrow x := x / y$ fi.
Let $\sigma = \{x = 3, y = 1\}$. Calculate $M(S, \sigma)$.

→ Here,
all the given guards are True
Therefore,
We have,

$M(S, \sigma) = \{\{x = 2, y = 1\}, \{x = 3, y = 2\}, \{x = 0, y = 1\}, \{x = 3, y = 1\}\}$

We got this by following guard execution.

Guard 1 : $x = x - 1 = 3 - 1 = 2$ & $y = 1$;
$\sigma = \{x = 2, y = 1\}$

Guard 2 : $y = y + 1 = 1 + 1 = 2$ & $x = 3$;
$\sigma = \{x = 3, y = 2\}$

Guard 3 : $x + y = 1 + 3 = 4 \Rightarrow x = 1/3 = 0$ & $y = 1$;
$\sigma = \{x = 0, y = 1\}$

Guard 4 : $x + y = 3 + 1 = 4 \Rightarrow x = 3/1 = 3$ & $y = 1$
$\sigma = \{x = 3, y = 1\}$

$\therefore M(S, \sigma) = \{\{x = 2, y = 1\}, \{x = 3, y = 2\}, \{x = 0, y = 1\}, \{x = 3, y = 1\}\}$

Date
Page

Q. 1

b) Let $w \equiv$ do $x > y \rightarrow x := x - 1 \ \square\ x > y \rightarrow y := y + 1 \ \square$
$x + 4 \rightarrow x := y / x \ \square\ x + y = 4 \rightarrow x := x/y$ od,
Let $\sigma = \{x = 3, y = 1\}$. Calculate $M(w, \sigma)$

→ Here,

Initial state : $\sigma = \{x = 3, y = 1\}$

For state 1 :

1. starting from $\{x = 3, y = 1\}$:
   Branch 1 : $x > y \rightarrow x := x - 1$
   condition : $3 > 1$ is true
   $\therefore x := 3 - 1 = 2 \Rightarrow$ New state : $\{x = 2, y = 1\}$
2. From $\{x = 2, y = 1\}$:
   Branch 1 : $x > y \rightarrow x := x - 1$
   condition : $2 > 1$ is true
   $\therefore x := 2 - 1 = 1 \Rightarrow$ New state : $\{x = 1, y = 1\}$
3. Final state : $\{x = 1, y = 1\}$

For state 2 :

1. starting from $\{x = 3, y = 1\}$
   Branch 1 : $x > y \rightarrow x := x - 1$
   condition : $3 > 1$ is true
   $\therefore x := 3 - 1 = 2 \Rightarrow$ New state : $\{x = 2, y = 1\}$
2. From $\{x = 2, y = 1\}$.
   Branch 2 : $x > y \rightarrow y := y + 1$
   condition $2 > 1$ is true
   $\therefore y := 1 + 1 = 2 \Rightarrow$ New state : $\{x = 2, y = 2\}$
3. From $\{x = 2, y = 2\}$:
   Branch 3 : $x + y = 4 \rightarrow x := y / x$
   condition : $2 + 2 = 4$ is true
   $\therefore x := 2/2 = 1 \Rightarrow$ New state : $\{x = 1, y = 2\}$
4. Final state : $\{x = 1, y = 2\}$

For state 2 :

1. starting from $\{x = 3, y = 1\}$:

   Branch 2 : $x > y \rightarrow y := y + 1$

   condition : $3 > 1$ is true

   $\therefore y := 1 + 1 = 2 \Rightarrow$ New state : $\{x = 3, y = 2\}$

2. From $\{x = 3, y = 2\}$:

   Branch 2 : $x > y \rightarrow y := y + 1$

   condition : $3 > 2$ is true

   $\therefore y := 2 + 1 = 3 \Rightarrow$ New state : $\{x = 3, y = 3\}$

3. Final state : $\{x = 3, y = 3\}$

For state 4 :

1. starting from $\{x = 3, y = 1\}$:

   Branch 1 : $x > y \rightarrow x := x - 1$

   condition : $3 > 1$ is true

   $\therefore x := 3 - 1 = 2 \Rightarrow$ New state : $\{x = 2, y = 1\}$

   ~~Branch 1~~ :

2. From $\{x = 2, y = 1\}$:

   Branch 1 : $x > y \rightarrow x := x - 1$

   condition : $2 > 1$ is true

   $\therefore x := 2 - 1 = 1 \Rightarrow$ New state : $\{x = 1, y = 1\}$

3. From $\{x = 1, y = 1\}$:

   Branch 1 : $x > y \rightarrow x := x - 1$

   condition : $1 > 1$ is false

   Branch 2 : $x > y \rightarrow y := y + 1$

   condition : $1 > 1$ is false

   Branch 3 : $x + y = 4 \rightarrow x := y / x$

   condition : $1 + 1 = 2$ i.e. false because its not 4

   Branch 4 : $x + y = 4 \rightarrow x := x / y$

   condition : $1 + 1 = 2$ i.e. false because its not 4

Date

Page

4. The loop terminates & we can now reduce x :

Keep executing the branches until $x = 0$ is reached. The resulting state becomes $\{x = 0, y = 1\}$

5. <u>Final state</u> : $\{x = 0, y = 1\}$

For state ⑤ :

It will be $\perp_d$ deadlock state which indicates that there are no further transitions available in the program.

$\therefore M(W, \sigma) = \{\{x = 1, y = 1\}, \{x = 1, y = 2\}, \{x = 3, y = 3\}, \{x = 0, y = 1\}, \perp_d\}$

Q. 2.

→ MAJORITY $\equiv k_0 = 0 \; ; \; k_1 = 0 \; ;$

while $k_0 < n \wedge k_1 < n$ do j ;

$k_0 := k_0 + 1$;

$k_1 := k_1 + 1$ od ;

if $k_0 = n \longrightarrow major := 1 \; \Box \; k_1 = n \longrightarrow major := 0$ fi

Here, $J \equiv$ do $b[k_0] = 1 \longrightarrow k_0 := k_0 + 1 \; \Box \; b[k_1] = 0 \longrightarrow$
$k_1 := k_1 + 1$ od.

Q. 3

a) If $M(S, \sigma)$ contains exactly one state, then S must be a deterministic statement.

→ $\boxed{\text{True}}$

Explanation: If the result of applying $M(S, \sigma)$ results in exactly one state, it indicates that there is no nondeterminism in the execution of S. Because of the key property of deterministic systems where, for every input, the system has only one possible next state.

b) If $\sigma \models \{p\}\ S\ \{q\}$, then $\sigma \models p$.

→ $\boxed{\text{False}}$

Explanation: If σ satisfies the triplet, then σ may or may not satisfies p. And we know that,

if $\sigma \not\models p$ then $\sigma \models \{p\}\ S\ \{q\}$

c) If $\sigma \not\models_{tot} \{p\}\ S\ \{q\}$, then $\sigma \not\models p$

→ $\boxed{\text{False}}$

Explanation: If $\sigma \not\models_{tot} \{p\}\ S\ \{q\}$ this does not typically mean or imply that $\sigma \not\models p$. Because the failure to prove q from p could occur for various reasons. Logically, if $\sigma \not\models_{tot} \{p\}\ S\ \{q\}$ then $\sigma \models p \wedge M(S, \sigma) \not\models q$

d) If $\sigma \models \{p\}\ S\ \{q\}$, then $M(S, \sigma) \models q$

→ $\boxed{\text{False}}$

Explanation: If $\sigma \models \{p\}\ S\ \{q\}$ & σ is partially correct, then $M(S, \sigma) - \perp \models q$. Therefore, $M(S, \sigma) \models q$ is incorrect & may contain pseudo states.

e) If $\sigma \not\models \{p\}\ S\ \{q\}$, then $\sigma \not\models_{tot} \{p\}\ S\ \{q\}$

→ $\boxed{\text{True}}$

Explanation:

$\sigma \not\models \{p\}\ S\ \{q\}$: If partial correctness does not hold, it means that the program may not lead to the correct postcondition q even if it terminates.

$\sigma \not\models_{tot} \{p\}\ S\ \{q\}$: Since total correctness includes partial correctness as part of the definition, it follows that total correctness also does not hold in this case

So, if $\sigma \not\models \{p\}\ S\ \{q\}$ then $\sigma \not\models_{tot} \{p\}\ S\ \{q\}$ is correct.

Q. 4.

a) $\{P(k, s+1)\}\ s := s + 1\ \{P(k, s)\}$

→ $\boxed{\text{True}}$

Explanation: Applying the backward assignment rule will satisfy the triplet

b) $\{P(k, s)\}\ s := s + 1\ \{P(k, s+1)\}$

→ $\boxed{\text{False}}$

Explanation: Let $\sigma = \{s = 1, k = 1\}$. We have $\sigma \models P(1, 1)$, but $M(s = s + 1, \sigma) = \{s = 2, k = 1\} \not\models P(1, 2)$

Date
Page

c) $\{P(k,s) \wedge s < 0\}$ $s := s + 1;$ $k := k + 1$ $\{P(k,s)\}$

→ $\boxed{\text{True}}$

Explanation: The above statement is True because the precondition is a contradiction & cannot be satisfied by any state.

d) $\{P(k,s) \wedge s = s_0\}$ $s := s + 1$ $\{P(k, s_0)\}$

→ $\boxed{\text{True}}$

Explanation: The precondition logically implies $P(k,s) \wedge P(k, s_0)$. Even after incrementing s, the post condition does not change, so $P(k, s_0)$ is still true.

e) $\{P(k+1, s+1)\}$ $s := s + 1;$ $k := k + 1$ $\{P(k,s)\}$

→ $\boxed{\text{True}}$

Explanation: Using backward substitution rule, we get,

$\{P(k+1, s)\}$ $k := k+1$ $\{P(k,s)\}$ and

$\{P(k+1, s+1)\}$ $s := s+1$ $\{P(k+1), s)\}$

Combine them using the sequence rule & we get:

$\{P(k+1, s+1)\}$ $s := s+1,$ $k := k+1$ $\{P(k,s)\}$.

Q. 5.

a) Let $\sigma \models \{x \neq 0\}$ while $x \neq 0$ do $x := x - 2$ od $\{x < 0\}$. what are the possible values of $\sigma(x)$?

→ Here,

possible values of $\sigma(x)$:

① When $\sigma(x) = 0$, then the precondition is not satisfied so the triple is satisfied

② When $\sigma(x) > 0$ and $\sigma(x)$ is even, S will terminate with some state τ with $\tau(x) = 0$, which ultimately does not satisfy the post condition.

③ When $\sigma(x) < 0$ or when $\sigma(x)$ is odd then S will diverge; which will be acceptable for partial correctness.

④ So under partial correctness, we got the equation $\sigma(x) \leq 0 \lor \sigma(x) \% 2 = 1$.

b) Let $\sigma \models_{tot} \{x \neq 0\}$ while $x \neq 0$ do $x := x - 2$ od $\{x < 0\}$, what are the possible values of $\sigma(x)$?

→ Here,

Since this is a total correctness, there cannot be any pseudo state involved in it. Therefore, no value of x can terminate the condition without giving divergence or error except when $x = 0$.

So, the only possible value for x in σ is 0 i.e. $\sigma(x) = 0$.

Date
Page

Q. 6. Let $\sigma \models_{tot} \{p_1\} S \{q_1\}$ and $\sigma \models_{tot} \{p_2\} S \{q_2\}$

a) $\{p_1 \wedge p_2\} S \{q_1 \vee q_2\}$

→ Here,

Precondition: The precondition $p_1 \wedge p_2$ is stronger than either p_1 or p_2 alone. It means that before executing S, both p_1 & p_2 must hold true.

Postcondition: Postcondition $q_1 \vee q_2$ is weaker than either q_1 or q_2 alone. We only require that either q_1 or q_2 holds after the execution of S.

Since both ~~$\{p_1\} S \{p_2\}$~~ $\{p_1\} S \{q_1\}$ & $\{p_2\} S \{q_2\}$ are correct under total correctness, this guarantees that S terminates when either p_1 or p_2 holds initially, however, we are assuming that both p_1 & p_2 hold initially. Thus, termination of S is still guaranteed under precondition $p_1 \wedge p_2$.

After executing S, since we know S will terminate & either q_1 or q_2 will be true based on initial correctness conditions of p_1 & p_2, we conclude that $q_1 \vee q_2$ will satisfied.

$\therefore \{p_1 \wedge p_2\} S \{q_1 \vee q_2\}$ is true under total correctness.

b) $\{p_1 \vee p_2\} S \{q_1 \wedge q_2\}$

→ Here,

Since $\{p_1\} S \{q_1\}$ and $\{p_2\} S \{q_2\}$ are correct under total correctness, we know that S will terminate when either p_1 or p_2 holds initially. This guarantees that S will always terminate when $p_1 \vee p_2$ holds.

Date
Page

If the precondition is $p_1 \vee p_2$ then :

i) If p_1 holds initially, we know that S will terminate & q_1 will hold, but we cannot guarantee that q_2 will hold.

ii) If p_2 holds initially, we know that S will terminate & q_2 will hold, but we cannot guarantee that q_1 will hold.

Therefore, we cannot guarantee that both q_1 & q_2 will hold after execution if only one of p_1 or p_2 is true initially.

$\therefore \{p_1 \vee p_2\}\ S\ \{q_1 \wedge q_2\}$ does not hold true under total correctness.

c) $\{p_1 \vee p_2\}\ S\ \{q_1 \vee q_2\}$

→ Here,

Since $\{p_1\}\ S\ \{q_1\}$ and $\{p_2\}\ S\ \{q_2\}$ are correct under total correctness, S will terminate if either p_1 or p_2 holds initially. Thus, termination of S is guaranteed under the precondition $p_1 \vee p_2$.

If p_1 holds initially, then by the total correctness of $\{p_1\}\ S\ \{q_1\}$, S will terminate and q_1 will hold, which satisfies $q_1 \vee q_2$. If p_2 holds initially, then by the total correctness of $\{p_2\}\ S\ \{q_2\}$, S will terminate and q_2 will hold, which also satisfies $q_1 \vee q_2$.

In either case, the postcondition $q_1 \vee q_2$ will be satisfied.

$\therefore \{p_1 \vee p_2\}\ S\ \{q_1 \vee q_2\}$ holds true under total correctness.

Date
Page

Q. 7. Let $\vdash \{p_1\} S \{q_1\}$ and $\vdash \{p_2\} S \{q_2\}$

a) $\{p_1 \wedge p_2\} S \{q_1 \wedge q_2\}$

→ Here,

Since $\{p_1\} S \{q_1\}$ holds, whenever p_1 is true before executing S, q_1 will hold afterward.

Similarly, since $\{p_2\} S \{q_2\}$ holds, whenever p_2 is true befor executing S, q_2 will hold afterward.

Given that both p_1 & p_2 are true initially because $p_1 \wedge p_2$ & since S guarantees that q_1 follows from p_1 & ~~p_2~~ q_2 follows from p_2, the postcondition $q_1 \wedge q_2$ will also hold.

∴ $\{p_1 \wedge p_2\} S \{q_1 \wedge q_2\}$ is valid under the partial correctness.

b) $\{p_2\} S \{q_1 \rightarrow q_2\}$

→ Here,

Since p_2 is true before execution, according to the given assumption $\{p_2\} S \{q_2\}$, q_2 will be true after executing S.

For Implication $q_1 \rightarrow q_2$:

If q_1 is true after executing S, since we know q_2 is also true after executing S, the implication $q_1 \rightarrow q_2$ holds true.

If q_1 is false after executing S, the implication $q_1 \rightarrow q_2$ also holds because a false antecedent makes the implication true regardless of the truth value of q_2.

Date________
Page________

$\therefore$ Since in both scenarios the implication $q_1 \rightarrow q_2$ holds, we can conclude that $\{p_2\}$ S $\{q_1 \rightarrow q_2\}$ is <u>valid</u> under partial correctness.

c) $\{\neg p_1 \rightarrow p_2\}$ S $\{\neg q_1 \rightarrow q_2\}$

→ Here,

to evaluate the validity, we consider the implication of the precondition :

if $\neg p_1$ is true, by the precondition $\neg p_1 \rightarrow p_2$, p_2 must be true before executing S. Since p_2 is true, by the assumption $\{p_2\}$ S $\{q_2\}$, we can conclude that q_2 will be true after executing S.

Considering q_1, the truth value of q_1 is not guaranteed from the precondition $\neg p_1 \rightarrow p_2$. If p_1 is true before executing S then next is implication :

Now,

Implication of postcondition $\neg q_1 \rightarrow q_2$:

If $\neg q_1$ is true ~~because~~ after executing S, from our analysis, if $\neg q_1$ is true we need to verify that q_2 is true. If p_2 is true, then q_2 is guaranteed to ~~the~~ be true from the assumption $\{p_2\}$ S $\{q_2\}$.

$\therefore$ $\{\neg p_1 \rightarrow p_2\}$ S $\{\neg q_1 \rightarrow q_2\}$ is <u>valid</u> under partial correctiness.

Date
Page

Q. 8.

a) $\models_{tot} \{w\}\ S\ \{q\}$

→ True

Explanation: The weakest precondition $wp(S, q)$ is specifically designed to ensure that the program S, when executed from a state satisfying w, will terminate & establish the post-condition q. Since the program is deterministic the total correctness will hold.

b) $\models \{w \wedge q\}\ S\ \{q\}$

→ True

Explanation: Here, $w \wedge q$ is a stronger precondition than w because w is the weakest precondition, also if all states totally satisfy a triple, then they also partially satisfy the triple.

c) There exists some state σ such that $\sigma \not\models w$ but $M(S, \sigma) \models q$.

→ False

Explanation: If $\sigma \not\models w$, there is no guarantee that executing S from σ will yield q. The definition of the weakest precondition w is meant to ensure that if it doesn't hold, we cannot assert anything about the validity of q after execution.

d) If $\sigma \models w$, then $M(S, \sigma) \neq \{\perp\}$

→ True

Explanation: $\sigma \models w$ means program S execute without undefined states. And $M(S, \sigma) \neq \perp$ is true as long as σ is well formed & satisfies w. So the above statement is True.

Date
Page

Q. 8.

e) If $\sigma \not\models w$, then $\sigma \models \{\neg w\}\ S\ \{\neg q\}$

→ $\boxed{\text{False}}$

Explanation: If $\sigma \not\models w$, it does not follow that $\sigma \models \{\neg w\}\ S\ \{\neg q\}$ is true. Just because the precondition does not hold does not guarantee that the postcondition will not hold. The behavior of the program S cannot be determined solely based on the failure of the weakest precondition.

Q. 9. Let $S \equiv y := y \% x$ and $q \equiv sqrt(y) > x$.

a) Calculate wlp(S, q).

→ Here,

$S \equiv y := y \% x$ and $q \equiv sqrt(y) > x$

The wlp rule for an assignment statement $y := E$ is

$wlp(y := E, q) \equiv q[y/E]$.

where, $q[y/E]$ means "substitute E for y in q"

In this case,

E is $(y \% x)$

so, we need to substitute $(y \% x)$ for y in q

$\therefore wlp(y := y \% x, sqrt(y) > x) \equiv$
$sqrt(y \% x) > x$

Therefore, the final result is:

$$\boxed{wlp(S, q) \equiv sqrt(y \% x) > x}$$

b) Calculate wp(S, q):

→ Here,

$S \equiv y := y \% x$ and $q \equiv sqrt(y) > x$

Now,

wp(S, q) for loop free programs:

Date
Page

$wp(S,q) \equiv D(S) \wedge wlp(S,q) \wedge D(wlp(S,q))$

$D(S) \equiv D(y := y \% x) \equiv x \neq 0$

$wlp(S,q) \equiv sqrt(y \% x) > x$

$D(wlp(S,q)) \equiv D(sqrt(y \% x) > x)$
$\equiv y \% x \geq 0 \wedge x \neq 0$

$$\boxed{\therefore wp(S,q) \equiv (x \neq 0) \wedge (sqrt(y \% x)) > x \wedge y \% x \geq 0.}$$

Q. 10. Let $S \equiv$ if $y \geq 0 \rightarrow x := y/x \;\square\; x \geq 0 \rightarrow x := x/y$ fi and $q \equiv x < y < z$.

a) Calculate $wlp(S,q)$

→ Here,
for the weakest liberal precondition (wlp) we want the weakest condition that must hold before exactly executing S such that if S terminates, q holds..

i) For branch $y \geq 0$, the postcondition becomes,
$$\frac{y}{x} < y < z$$

because after $x := y/x$, x is replaced by y/x in postcondition.

ii) For branch $x \geq 0$, the postcondition becomes,
$$\frac{x}{y} < y < z$$

Because after $x := \frac{x}{y}$, x is replaced by $\frac{x}{y}$ in the postcondition.

$$\therefore wlp(s, q) = (y \geq 0 \rightarrow y/x < y < z) \wedge (x \geq 0 \rightarrow x/y < y < z)$$

b) Calculate $wp(s, q)$

→ Here,

$$D(s) \equiv (y \geq 0 \rightarrow x \neq 0) \wedge (x \geq 0 \rightarrow y \neq 0)$$

step i) :

$$y \geq 0 \rightarrow x \neq 0$$

$$\therefore wp(x := y/x\}, q) \equiv \left(\frac{y}{x} < y < z\right)$$

step 2:

$x \geq 0 \rightarrow y \neq 0$

$\therefore wp(x := x/y;\ q) \equiv \left(\frac{x}{y} < y < z\right)$

$D(wlp(s, q)) \equiv x \neq 0 \wedge y \neq 0$

$\therefore wp(s, q) \equiv \left(y \geq 0 \rightarrow \left(x \neq 0 \wedge \frac{y}{x} < y < z\right)\right) \wedge$

$\left(x \geq 0 \rightarrow \left(y \neq 0 \wedge \frac{x}{y} < y < z\right)\right) \wedge$

$(x \neq 0 \wedge y \neq 0)$